

DISTRIBUTING DATACAST SIGNALS EMBEDDED IN BROADCAST
TRANSMISSIONS OVER A COMPUTER NETWORK

5

Jordan Du Val

Wen Li

COPYRIGHT NOTICE

- 10 A portion of the disclosure of this patent document contains material which is subject to copyright protection. The copyright owner has no objection to the facsimile reproduction by anyone of the patent document or the patent disclosure, as it appears in the Patent and Trademark Office patent file or records, but otherwise reserves all copyright rights whatsoever.

CROSS-REFERENCE TO RELATED APPLICATION

- 15 This application is a continuation-in-part of the U. S. application entitled "PERSONAL COMPUTER USED IN CONJUNCTION WITH TELEVISION TO DISPLAY INFORMATION RELATED TO TELEVISION PROGRAMMING," serial no. 09/585,266, filed on May 30, 2000, which is hereby incorporated by reference in its entirety.

- 20 CROSS-REFERENCE TO CD-ROM APPENDIX AND APPENDIX A

- CD-ROM Appendix A, which is part of the present disclosure, is a CD-ROM appendix consisting of 430 files. CD-ROM Appendix A is a computer program listing appendix that includes a software program. The total number of compact disks including duplicates is two. Appendix B, which is part of the present specification, contains a list
25 of the files contained on the compact disk. Appendix A and Appendix B are incorporated

herein by reference. The attached CD-ROM Appendix A is formatted for an IBM-PC operating a Windows operating system.

FIELD OF THE INVENTION

This invention relates to interactive audio and visual entertainment, such as live or
5 recorded interactive television programming, and other interactive audio and video
content. In particular, this invention relates to systems and methods for distributing
interactive data extracted from audio-visual content to a plurality of users over a
computer network.

BACKGROUND

10 The distribution of enhanced television content to a plurality of users via
commercially available set top boxes is known. In one system, a set top box is connected
to a television and to the Internet. The set top box receives signals embedded in the
television signal's vertical blanking interval (VBI) and extracts the enhanced television
content encoded in the signals. The signals may be in accordance with the Advanced
15 Television Enhancement Forum (ATVEF) Enhanced Content Specification, a well-known
industry standard. (Additional information relating to ATVEF standards may be obtained
from the Internet at <http://www.atvef.com>.) In a typical application, the enhanced
television content includes a URL identifying the location of a computer resource on the
Internet—typically a remote server system—along with a short description, such as a text
20 label, of the information and processing supported by the computer resource. The
enhanced television content is generally synchronized with the television content, such as
a commercial advertisement, and thus provides access via the Internet to supplemental
information and processes relating to the television content (hereafter, "supplemental
processing") contemporaneously with the user's viewing of the television content.

25 Fig. 1 is a time-sequence diagram illustrating the synchronization of interactive
data (e.g., enhanced television content) with video content (e.g., television programming)
in the prior art. In Fig. 1, portions of a video signal 4 and a stream of interactive data 6
are shown occurring over five time intervals A-E measured over time-line 2. The video
signal 4 includes video content 6A and 6B (e.g., a television sit-com), interspersed by

three commercial advertisements 8, 10, and 12. A stream of interactive data 6 includes five series of interactive data 0-4 synchronized with the occurrence of the sequence of program content and commercial advertisements 6A, 8, 10, 12, and 6B respectively. Commercial content 2 10, for example, may include a television advertisement for Starbucks's brand coffee 10A occurring over a 30-second interval 16 (time interval C). Interactive data 2 22 synchronized with commercial content 2 10 (time interval C) thus typically relates to the Starbucks's brand of coffee. The enhanced television content 2 22 may thus include the name of the location of a remote computer resource on the Internet, such as "http://www.starbucks.com," supporting on-line processes supplementing the Starbucks's brand coffee advertisements (e.g., advertisement promotions, e-commerce transactions).

In typical applications, therefore, the set top box extracts a sequence of interactive data from the video signal for display to the user. The user may then select a remote computer resource identified in the interactive data, causing the set top box to access the supplemental processing on the remote computer resource for display to the user on the television.

Although this technique achieves good results, it requires a special set top box or a special television tuner card for use with a personal computer. It is desirable, however, to distribute interactive data embedded in video and audio content to a plurality of users using conventional personal computing devices without additional hardware, including mobile devices which are often limited in expandability. It is also desirable to distribute interactive data using a scalable processing architecture capable of handling synchronous distribution of large volumes of interactive data. This would make the experience of interactive data more convenient for the mobile user as well as reduce the cost of the system for any user.

SUMMARY

A system and method is described for distributing interactive data extracted from a video signal encoding video content to a plurality of client computers via a computer network. The interactive data is distributed to the user contemporaneously with the user's experience of the encoded video content. In some embodiments, a plurality of data

source computers extract the interactive data from the video signals and forward them to a distribution server. In some embodiments, the distribution server buffers the interactive data and broadcasts the interactive data to a Web server cluster. In some embodiments, a program executing on each client computer periodically sends updation requests to the Web server cluster to retrieve new interactive data for display to the user. In some embodiments, a re-direct server receives a user request for access to a remote computer resource identified in the interactive data and re-directs the user request to the remote computer resource. In some embodiments, a computer program operating within a Web server and the distribution server further enables script files containing additional interactive data to be created and processed by the system.

BRIEF DESCRIPTION OF THE DRAWINGS

Fig. 1 is a time-sequence diagram illustrating interactive data synchronized with a video signal in the prior art.

Fig. 2 is a flow diagram illustrating a method of distributing live data extracted from a video signal to a plurality of client computers, according to some embodiments of the present invention.

Fig. 3 is a screenshot of the distributed live data of Fig. 2 displayed on a client computing device compatible with some embodiments of the present invention.

Fig. 4 is a block diagram illustrating the network components of a System compatible with some embodiments of the present invention.

Fig. 5A is a block diagram illustrating the data flow between the System network components of Fig. 4, according to some embodiments of the present invention.

Fig. 5B is a flow diagram illustrating the process stages of the data flow of Fig. 5A, according to some embodiments of the present invention.

Fig. 6 is a flow diagram illustrating additional process stages performed by the live data source computers, according to some embodiments of the present invention.

Fig. 7 is a block diagram illustrating a data structure for an interactive event compatible with the present invention.

5 Figs. 8A-8B illustrate in more detail the logic flow and process stages performed by the distribution server as described in Fig. 5B (stages 158-162), according to some embodiments of the present invention.

Figs. 9A-9B illustrate in more detail the logic flow and process stages performed by the Web servers in the Web server cluster 104 as described in Fig. 5B (stages 164-166), according to some embodiments of the present invention.

10 Fig. 10 is a flow diagram illustrating in more detail the process stages performed by the client computer, according to some embodiments of the present invention.

Fig. 11 is a block diagram illustrating the processing of script events in combination with the processing of live events by the System, according to some embodiments of the present invention.

DETAILED DESCRIPTION

15 As used herein, "video content" shall refer to content generated during the performance of an audio-visual work. Unless otherwise noted, the term "video content" includes audio-only content (e.g., a radio program), video-only content (e.g., silent motion picture, or a silent motion picture with captions), or any combination of audio-, video-, or other content that one skilled in the art would understand as compatible with
20 the present invention.

As used herein, "live data" is interactive data synchronized with the performance of video content. The interactive data may be extracted from a broadcast transmission, or additionally extracted from a stored medium, such as a DVD, video cassette, or audio recording (note, therefore, that "live data" does not mean a live performance). "Script
25 data" shall refer to interactive data that is not synchronized with the performance of video content.

In some embodiments of the present invention, at least one server computer connected to a plurality of client computing devices is programmed to perform the process stages illustrated in Fig. 2. In a first stage 40, the server computer processes a series of live data. In a second stage 42, the server computer distributes the series of live data synchronously to a plurality of client computers over a computer network, such as the Internet. Because the distribution is synchronous, the live data is processed and distributed to the client computing devices within a time period short enough to ensure that the user's experience of the live data (using the client device) is contemporaneous with the user's experience of the video content (the video content being rendered using any conventional content display device, e.g., a television, radio, PDA, computer, including the client computer); in some embodiments, this time period (hereafter "response time") is no longer than 15 seconds.

Fig. 3 is a screenshot of the distributed live data of Fig. 2 displayed on a client computing device compatible with some embodiments of the present invention. In this embodiment, the series of interactive data is displayed as a scrolling list of conventional text hyperlinks 50 updated within a window 54. An identification 52, such as a label, icon, or logo, of the carrier of the video content is additionally displayed. Each new live data distributed to a particular client device is distributed by the server computer to the client device as a new text hyperlink, e.g., 56, to be posted in the list 50. The distribution is designed so that the interactive data is posted to the user on the client device within the response time, assuming no occurrence of unrelated processing errors (e.g., a network communication error).

Fig. 4 is a block diagram illustrating the network components of a system compatible with some embodiments of the present invention. In Fig. 4, at least one first server computer 102 is connected to a cluster of second server computers 104, both of which are in turn connected to a third server computer 106 controlling a storage device. The first, second and third server computers 102, 104, and 106 are programmed to process data and instructions comprising the various embodiments of the present invention; the server computers 102, 104, and 106 properly programmed to perform the operations of the various embodiments of the present invention are hereafter referred to as the "distribution server," "Web server cluster" (or, singly, "Web server") and "database

server" respectively. The terms "distribution server," "Web server cluster" (or, singly, "Web server") and "database server" refer herein to the programmed computers, i.e., to both the hardware and software components, unless otherwise stated or implied by context. The "distribution server" 102, "Web server cluster" 104, and "database server" 106 shall be collectively referred to as the "System servers" 100. First, second and third computers 102, 104, and 106 (i.e., "distribution server" 102, "Web server cluster" 104, and "database server" 106 respectively) generally include any conventional general-purpose server computers. In some embodiments, the second computers (hosting the Web server cluster 104) include Supermicro SuperServers 6010H, manufactured by Supermicro, Inc. of San Jose, California, equipped with two Intel 700 MHz CPUs, 512 MB of RAM, 9GB SCSI hard drives, and conventional NICs, among other standard components. In some embodiments, the first and third computers include a Caliber CP2700, manufactured by Caliber Corporation of Fremont, California, equipped similarly to the Supermicro SuperServer 6010H.

It should be noted that a single computer may be programmed to perform the operations of the distribution server 102, Web server 104, and database server 106, and therefore the latter terms refer primarily to the computational processes constituting the present invention, and not the particular hardware implementation of a subset of such processes. For example, in some embodiments, database server and the distribution server are hosted on the same machine, thus sharing the same processor. It should additionally be noted that by using conventional distributed programming techniques, the number of server computers needed to optimally implement the various embodiments of the present invention will vary depending upon the amount of computer resources required to support the use of the System (i.e., generally a function of the quantity of interactive data processed and distributed to users). In this disclosure, a typical embodiment of the System 100 is described.

System servers 100 are in turn connected to a plurality of computers 112 via computer network 98. The computer network 98 may include the Internet, a local area network (LAN), a wide area network (WAN), a metropolitan area network (MAN), an interactive television network, a wireless network, and generally any other connection capable of delivering electronic content between computer devices. The plurality of

computers 112 are programmed to receive video signals from a number of carriers over conventional transmission media (e.g., satellite, cable and air), extract the live data from video signals, and forward the live data in real-time to the distribution server 102. Each carrier may transmit a different video signal for each of a predetermined number of time zones; in some embodiments, one of the plurality of computers 112 will be assigned to receive each of the video signals for the carrier. The plurality of computers 112 programmed in accordance with the present invention are hereafter referred to as “live data sources” or “live data source computers.” The live data sources 112 are implemented using conventional general-purpose computers well-known to those skilled in the art. In particular, each of the plurality of computers 112 are in some embodiments equipped with a conventional Hauppauge TV card for capturing and extracting the interactive data from the live video signal.

System servers 100—in particular, via Web server cluster 104—communicate with a plurality of users 114 operating client computers 116 via computer network 98.

Client computers include conventional general-purpose computers typically used by users, such as a standard notebook or desktop computer, as well as more specialized computing devices, such as the various consumer mobile devices (e.g., PDA, cell phone). In general, client computers include any computing device capable of performing data communication with computer resources (network servers) via the computer network 98.

Client computers 116 and System servers 104 are additionally connected to a plurality of remote computer resources 120 via computer network 98. Computer resources 120 include, in some embodiments, the millions of remote computer systems interconnected by computer network 98. System servers 100—in particular, via distribution server 102—communicate with at least one non-synchronous interactive data (script data) producer 122 (hereafter, “data producer”) operating one of the plurality of client computers.

Figs. 5A and 5B illustrate the data flow within the System 100, according to some embodiments of the present invention. Fig. 5A is a block diagram illustrating the data flow between the System network components of Fig. 4, and Fig. 5B is a flow diagram illustrating the process stages of the data flow of Fig. 5A. The two figures—Figs. 5A and 5B—are described together. In stage 152, each of the plurality of live data source

computers 112 receive a video signal via conventional broadcast transmission (e.g., cable, satellite, air). In stage 154, the live data source computers 112 extract the live data from the video signals, and then (stage 156) forward a stream of the newly extracted live data to the distribution server 102. In stage 158, the distribution server 102 buffers the incoming live data from the data source computers 112. In stage 160, the distribution server 102 stores a record of the incoming live data, and (stage 162) broadcasts the live data to the Web server cluster 104. In stage 164, the Web server cluster 106 receives a plurality of requests for the most current live data (hereafter "update request") from a plurality of client computers 116. In stage 166, the Web server cluster 104 processes the update requests, and sends the most current interactive data to the requesting client computer. In stage 168, the re-direct server 108 receives a plurality of user requests to access a remote computer resource 120. In stage 170, the re-direct server re-directs the user request to the computer resource 120 for processing by the computer resource 120, typically a network server. In stage 172, the re-direct server stores a record of the re-directed user request in the database server 106. As illustrated by dotted boxes 174-180, the various stages within each of the dotted boxes are performed by the live data source computers 112, distribution server 102, Web server cluster 104, and remote computer resources 120 respectively.

Fig. 6 is a flow diagram illustrating additional process stages performed by the live data source computers 112, according to some embodiments of the present invention. Prior to performing the process stages described in box 174, each local data source computer 112 in stage 202 opens a conventional socket connection using an available pre-determined port with the distribution server 102 over computer network 98. The live event 220 is forwarded to the distribution server 102 using a customized application-level protocol—instead of the conventional HTTP—built on top of the conventional TCP transport protocol. A customized protocol is used to maximize the throughput and bandwidth for forwarding the interactive events to the distribution server by avoiding excessive processing overhead arising from the use of the HTTP protocol. For example, because HTTP is generally designed by default to close the socket link after each data transfer, processing time (and therefore communication bandwidth) is wasted by having to re-open the socket link after each data transfer, or by having to execute an additional instruction to keep the socket link open.

In some embodiments, the live data source computer 112 then performs stages 152-154 as described in Fig. 5B. In the next stage (stage 204), the local data source tags each live data extracted from a video signal with data uniquely identifying each live data; in some embodiments, this additional identifying data includes a timestamp and an extended carrier ID which uniquely identifies the respective carrier of the video signal, and the time zone to which the video signal is directed. As a result of the tagging performed in stage 204, a data structure is created using the interactive data; the data structure thus created is hereafter referred to as an "interactive event" or an "event." Fig. 7 is a block diagram illustrating a data structure for an event compatible with the present invention. In some embodiments, the event 220 data structure includes an extended_carrier_ID 222, a URL 224, a label 226 typed as strings, and a timestamp 228 typed as a long integer. After the live data source computer 112 constructs each extracted live data into a live interactive event 220 ("live event"), the live data source computer 112 immediately forwards the live event 220 to the distribution computer over the opened socket connection.

In some embodiments, the processes used for capturing and forwarding the interactive events executing on the live data source computers 112 are coded in C, compiled and run as a stand-alone application within a Red Hat Linux operating environment (Red Hat Linux V6.2) well-known to those skilled in the art. (The Red Hat Linux operating system is manufactured by Red Hat Corporation of Durham, North Carolina.)

Figs. 8A-8B illustrate in more detail the logic flow and process stages performed by the distribution server 102 as described in Fig. 5B (stages 158-162), according to some embodiments of the present invention. In stage 240, the distribution server 102 opens a first socket connection (in some embodiments, over port 2000) to each of the live data source computers 112. In stage 242, the distribution server spawns a live event capture thread 270 to listen on the first socket connection for new live events 220 received from the data source computers 112. In stage 244, the distribution server 102 posts (buffers) 272 the new live events 220 to a central distribution queue 274. In stage 246, the distribution server 102 opens a second socket connection (in some embodiments, over port 2001) to the Web server cluster 104. In stage 248, the distribution server 102 spawns

a distribution thread 276 which periodically retrieves 278 the new live events 220 (and script events, which are discussed below in reference to Figs. 5A and 12) from the central distribution queue 274 for broadcasting 280 over the second socket connection to the Web server cluster 104. In stage 250, the distribution server broadcasts the live events
5 (and script events as described in reference to Fig. 5A and 13) to the Web server in the Web server cluster 106 over the second socket connection opened in stage 244.

In some embodiments, the live events are broadcast over the socket connection using the same customized application protocol used for communications between the distribution server 102 and the live data source computers 112 (described in reference to
10 Fig. 6, stage 202). The customized protocol provides more efficient communication of events to the Web server cluster 104 than is obtainable using standard HTTP, which is critical for enabling the System 100 to distribute events within the desired response time. In stage 252, the distribution server 102 flushes the central distribution queue of the events broadcast in the previous stage (250). In stage 254, the distribution server 102
15 spawns a recordation thread to automatically identify 294 new events, and to store 292 a record of each new event in the database server 106. The process stages described in Fig. 8B may be performed in various orders; for example, the distribution server 102 may spawn the threads in stages 242, 248 and 252 in any order during start-up of the System 100. Distribution server 102 provides required efficiency and bandwidth to the System
20 100 enabling it to distribute large numbers of events to large numbers of user within the required response time. In particular, in some embodiments, live data source computers 112 may be limited in their programming to the transmission of the live events to a single IP address, i.e., a single host computer; in these embodiments, it is overly expensive to re-program the live data source computers 112 to provide multi-connection capability. In
25 addition, without distribution server 102, each Web server 104 would need to execute processes for listening and receiving data from the multiple live data source computers 112. This added processing requirement will unsatisfactorily slow the ability of the Web servers to efficiently respond to user updation requests within the required response time, especially as usage of the System 100 increases.

30 In some embodiments, the processes performed by the distribution server 102 are coded in Java, compiled into Java bytecodes, and executed within a Java Virtual Machine

well-known to those skilled in the art. In some embodiments, the Java Virtual Machine runs within a Windows 2000 Server operating system also well-known to those skilled in the art.

5 Figs. 9A-9B illustrate in more detail the logic flow and process stages performed by the Web servers in the Web server cluster 104 as described in Fig. 5B (stages 164-166), according to some embodiments of the present invention. In general, in some embodiments, the Web servers constituting the Web server cluster 104 are programmed similarly to perform the operations described in Fig. 9; thus, although the following process stages are described in reference to a single Web server, they are generally
10 applicable to each Web server in the Web server cluster 104. In stage 280, a cluster 302 of carrier distribution queues 302A-302n are created within the Web server in which a single carrier distribution queue 302A-302n is assigned to each of a predetermined number of time zones for each event carrier (i.e., in some embodiments, one carrier distribution queue is created for each unique extended_carrier_ID). In stage 282, the
15 Web server 104 opens a conventional socket connection (in some embodiments, over port 2001) with the distribution server 282. In stage 284, the Web server spawns a listening thread 300 for receiving new events broadcast over the socket connection from the distribution server 102.

In stage 286, the new events received from the distribution server are identified by
20 carrier and time zone (i.e., extended_carrier_ID), and posted 304 to the appropriate carrier distribution queue 302A-302n (i.e., the queue corresponding to the carrier and time zone of the event). In stage 288, the Web server spawns an updation processing thread 308 to process conventional HTTP (Hyper-text Transport Protocol) requests 312 received from users requesting new events (i.e., updation request). In stage 290, the Web
25 server receives an updation request 312 from a client computer 116, which includes as parameters data identifying a carrier and a time zone (the time zone being retrieved from the cookie file associated with the user), and the timestamp of the most current event received by the tuner. In stage 292, the Web server 104 identifies and retrieves 310 the new events from the relevant carrier distribution queue 302A-302n using the parameter
30 information; in some embodiments, for example, the Web server will determine the appropriate carrier distribution queue by mapping the data identifying the carrier and time

zone into a corresponding extended_carrier_ID, and then search through appropriate carrier distribution queue comparing the timestamps of the queued events with the timestamp of the most current event received from the tuner. All of the queued events, therefore, that have timestamps later in time to the timestamp of the most current event on the tuner are thereafter in the next stage 294 sent to the tuner by the Web server 104.

In some embodiments, any general-purpose Web server 104 may be used to implement the various embodiments of the presenting invention. In some embodiments, for example, Web server 104 includes the Microsoft Internet Information Server 5.5, manufactured by Microsoft Corporation of Redmond, Washington, executing within a Microsoft 2000 Server operating environment, also manufactured by Microsoft Corporation. In some embodiments, application logic for the processes described in reference to Figs. 9A-9B are coded as one or more Java servlets. In some embodiments, the servlets are executed within a commercial servlet container (not shown), such as the BEA Weblogic 5.1 application server, manufactured by BEA Corporation of San Jose, California. Web server 104 thus processes HTTP requests received from the client computers 116 by invoking servlet processes from the application server. The Weblogic application server additionally includes built-in distributed processing, load-balancing, and clustering capabilities enabling a Web server cluster to be efficiently created from individual Web servers. The use of servlets and servlet containers for coding Web server logic is well-known to those skilled in the art. Additional information describing the use and operation of Java servlet and BEA Weblogic application server technologies are available over the Internet at <http://www.sun.com> and <http://www.bea.com> respectively.

Fig. 10 is a flow diagram illustrating in more detail the process stages performed by the client computer, according to some embodiments of the present invention. Stage 320-330 describe processes performed by the client computer 116 upon access to the System 100 by a user. Stages 332-344 describes processes performed by the client computer 116 after it has accessed the System 100. In stage 320, a user initiates usage of the System 100 via the client computer 116 by executing a small program (hereafter "tuner" or "tuner program") (not shown) in the local address space of the client computer 116. The tuner may be made available for execution by the user using a number of conventional techniques. For example, in one embodiment, the tuner may be uploaded as

an applet into the client computer 116 from the Web server cluster in response to an initial HTTP request to access the System 100; in another embodiment, the tuner may be downloaded via the computer network 98 as a binary file for stand-alone execution within a particular operating environment, such as a Microsoft Windows operating environment; in yet another embodiment, the tuner may be coded in Javascript, embedded in the HTML pages, and processed by a Java-enabled Web browser (i.e., Java Virtual Machine) during processing of the HTML pages. In general, the tuner program must be capable of establishing data communication with the Web server cluster using conventional Web communication protocols (i.e., HTTP over TCP/IP using, typically, public port 80).

10 In stage 322, the tuner program creates a user event queue. In stage 324, the tuner program identifies the user using a conventional cookie file previously stored in the local file system of the client computer 116; in some embodiments, the user is identified by the user's email address previously submitted by the user during a registration process. If a cookie is not found, the cookie may at this stage be re-created using data stored in the database server 106; if no data (e.g., the user's email address) is stored in the database server 106 identifying the user, then the user may be required to enter a registration process with the System 100 for collection of this information. In stage 326, the tuner program identifies the time zone of the user as identified in the cookie. In stage 328, the tuner program sends a carrier change request to the Web server cluster 104 over computer network 98 using HTTP. The tuner sends a carrier change request either upon initial access to the System 100 or in response to a user selection to receive events from a different carrier. The carrier change request includes data identifying the user time zone and a user selected carrier; in some embodiments, a default carrier may be predetermined for the initial carrier change request. In stage 330, the tuner program receives a response from the Web server cluster 104 which includes the latest events required to populate the user event queue for the carrier specified by the user (or as specified by default by the tuner program) in the previous stage.

After initial execution of the tuner program, then in stage 332 the tuner program periodically sends an updation request over HTTP to the Web server cluster 104 to receive relevant new events. The periodic requests sent by the tuner program are hidden from the user and enable the System 100 to distribute new events 220 to the user in

pseudo-push fashion. In some embodiments, the updation requests are sent by each client computer every 7.5 seconds. In these embodiments, 7.5 seconds represents the Nyquist sampling frequency (generally half of the duration of the minimum target sample) for a 15-second video signal constituting the typically shortest commercial advertisement used by carriers, e.g., commercial content 1 in 15-second time interval B (Fig. 1). The updation request period however may be adjusted to any time interval depending upon a number of factors, e.g., the particular video content (live events for game shows may require short intervals—users may be “participating” in the game show in real-time using a remote computer resource), and the bandwidth limitations of the System 100 (millions of users sending requests over a short period of time may congest the Web server cluster’s 104 ability to process the requests), among other considerations. In stage 334, the tuner program receives the new events 220 in response to the updation request.

In stage 336, the tuner program updates the user event queue with the newly received event 220. In stage 338, the tuner program displays the new events to the user, as, for example, illustrated in Fig. 3. In stage 340, the tuner program receives a selection of a remote computer resource (via, e.g., selection of a hyperlink encoded with the URL for the remote computer resource), such as network server 130 (Fig. 4) by the user. In stage 342, the tuner program sends an HTTP re-direct request to the re-direct server 108 which includes the location of the (typically remote) computer resource (i.e., the URL of the computer resource) selected by the user. A conventional Web browser (not illustrated) running on the client computer then receives the response directly from the selected remote computer resource by the user. The Web browser may include Internet Explorer, manufactured by Microsoft Corporation of Redmond, Washington, or Netscape Navigator, manufactured by Netscape Communications Corporation of Mountain View, California.

The re-direct server 108 includes a conventional Web server programmed to collect information relating to user activities in response to event selections, and to store the information in the database server 106. User activity information stored in the database server 106 includes a record of each re-direct request, including the location of the selected remote computer resource and the IP address of the client computer 116 used by the user. The user activities collected by the re-direct server 108 enables—in

conjunction with profile information collected from the user during, e.g., a user registration process—enables the System 100 owner to generate reporting information supporting customer relationship management, decision-making and other business needs of the owner.

5 Distribution server 102, Web server cluster 104, and re-direct server 108 communicate with database server 106 using conventional techniques. Database server 106 is hosted by any suitable general-purpose server computer well-known to those skilled in the art, such as the Caliber CP2700, manufactured by Caliber Corporation of Fremont, California. Any robust commercial database management system software may
10 be used to implement database 106. In some embodiments, the database management system software includes Microsoft SQL Server Version 7.0 manufactured by Microsoft Corporation. Network communication with the database server 108 by the distribution server 102, Web server cluster 104, and re-direct server 108 is performed using appropriate database driver software loaded into the distribution server 102, Web
15 server cluster 104, and re-direct server 108 respectively. Appropriate database drivers for Microsoft SQL may be downloaded from the Internet at <http://www.microsoft.com>.

Fig. 11 is a flow diagram illustrating the processing of script links by the System 100, according to some embodiments of the present invention. In some embodiments, script link processing by the System 100 is accomplished using software having a front-
20 end component and a back-end component. In some embodiments, the front-end component includes a script event generation program (hereafter “event generation program”) 102-E (Fig. 5A) uploaded into a conventional Web server 102-W hosted on computer 102 (i.e., along with the distribution server 102. Web server 102-W may, however, be hosted on any properly interconnected and configured server computer. The
25 event generation program 102-E provides a Web-based interface for access by an interactive data producer 122 via a client computer 116 (Fig. 4) over the computer network 98. The event generation program 102-E generates a series of webpages enabling the interactive data producer 122 to enter one or more individual script events for automatic assembly into a script file readable by the back-end script component. The
30 event generation program 102-E also enables a user to submit an already assembled script file containing a series of script events for processing by the back-end component.

In some embodiments, the back-end component includes a script processing program 102-P uploaded into the distribution server enabling the distribution server to distribute events assembled in a properly formatted script file. In particular, in some embodiments, distribution server 102 spawns a script directories management thread 440 which checks the headers of pending script files (stored, e.g., in the local file system of the host computer) to determine the time when they are to be distributed to the client computers 116. When a script file is determined to be ready for distribution, the script directories management thread 440 spawns a script process thread 442 to retrieve the script events from the script file. The script directories management thread 440 then posts the script links retrieved from the script file to the central distribution queue 274. The script directories management thread 440 additionally notifies the recordation thread 290 to store a record of the script event activities within the database server 106.

Fig. 12 is a block diagram illustrating the processing of script events in combination with live events by the System 100, according to some embodiments of the present invention. An exemplary portion of a script file 400 is illustrated containing five script events S6-S9. Portions of two exemplary series of live events are also illustrated: a first portion 402 received from a carrier 1 408, and a second portion 404 received from a carrier 2 410. Each portion 402-404 includes live events B3-B6 and C11-C14 respectively. Distribution server 102 receives live events 402-404 from live event source computers 112, and receives script events 400 from script processing program 102-P executing within the distribution server 102. Distribution server stores the processed script events 400 and the live events 402-404 in the central distribution queue 274, and then broadcasts the stored events 400-404 to the Web server cluster 104. In general, the Web server cluster 104 and tuner programs 410 process the script events and live events identically. Accordingly, depending on how the script events are identified in the script file—i.e., in some embodiments by extended carrier ID—the Web server cluster 104 will appropriately send the script event to the appropriate corresponding carrier distribution queue, e.g., 302A, in accordance with stage 286 (Fig. 9B). As additionally illustrated by “virtual” carrier queue 406 in Web server 104, one or more “virtual carriers” can be created and maintained by the System 100 (the virtual carrier is, e.g., assigned a unique extended carrier ID 222). A virtual carrier as used herein refers to a “carrier” of script events unrelated to any contemporaneous performance of video or other content. Note

that although virtual script events are non-synchronous, script events can be created to be synchronized with video content. This is illustrated by carrier queue 302B—in which script events S5 and S6 were created to be included within the series of live events 404 between the occurrences of C12 and C13; this is additionally illustrated by user
5 distribution queue 414 containing script events S5 and S6 already distributed to the user client computer 116.

Although various embodiments of the invention have been shown and described, the invention is limited only by the following claims.

APPENDIX A

See attached CD-ROM Copy 1 or Copy 2

APPENDIX B

Volume in drive E is 010918_1125

Volume Serial Number is CBDE-642F

Directory of e:\SPOTNE~1

```

5  09/18/01    11:25a    <DIR>      .
    09/18/01    11:25a    <DIR>      ..
    09/18/01    11:25a    <DIR>      DATABASE
    09/18/01    11:25a    <DIR>      DISTRI~1
    09/18/01    11:25a    <DIR>      INTERA~1
10 09/18/01    11:25a    <DIR>      KAPRON
    09/18/01    11:25a    <DIR>      ON-LIN~1
    09/18/01    11:25a    <DIR>      REDIRE~1
    09/18/01    11:25a    <DIR>      TUNER
    09/18/01    11:25a    <DIR>      TWWEBS~1
15                                10 File(s)      0 bytes

```

Directory of e:\SPOTNE~1\DATABASE

```

20 09/18/01    11:25a    <DIR>      .
    09/18/01    11:25a    <DIR>      ..
    09/18/01    11:25a    <DIR>      TWDB
                                3 File(s)      0 bytes

```

Directory of e:\SPOTNE~1\DATABASE\TWDB

```

25 09/18/01    11:25a    <DIR>      .
    09/18/01    11:25a    <DIR>      ..
    09/18/01    11:25a    <DIR>      DOCS
    09/18/01    11:25a    <DIR>      INSTAL~1
    09/18/01    11:25a    <DIR>      STORED~1
    09/18/01    11:25a    <DIR>      TABLES
30 09/18/01    11:25a    <DIR>      TRIGGERS
                                7 File(s)      0 bytes

```

Directory of e:\SPOTNE~1\DATABASE\TWDB\DOCS

```

35 09/18/01    11:25a    <DIR>      .
    09/18/01    11:25a    <DIR>      ..
    09/14/01    10:01a             702 README.TXT
    09/14/01    10:01a             48 VSSVER.SCC
                                4 File(s)      750 bytes

```

Directory of e:\SPOTNE~1\DATABASE\TWDB\INSTAL~1

```

40 09/18/01    11:25a    <DIR>      .
    09/18/01    11:25a    <DIR>      ..
    09/14/01    10:01a             407 INSTALL.BAT
    09/14/01    10:01a             1,778 INSTAL~1.BAT
45 09/14/01    10:01a             1,062 INSTAL~2.BAT
    09/14/01    10:01a             80 VSSVER.SCC
                                6 File(s)      3,327 bytes

```

```

Directory of c:\SPOTNE~1\DATABASE\TWDB\STORED~1
09/18/01    11:25a    <DIR>
09/18/01    11:25a    <DIR>
5  09/14/01    10:01a    3,792 SP0B71~1.SQL
    09/14/01    10:01a    1,541 SP14A0~1.SQL
    09/14/01    10:01a    5,122 SP1735~1.SQL
    09/14/01    10:01a    2,992 SP66D4~1.SQL
    09/14/01    10:01a    3,749 SPD0F2~1.SQL
10 09/14/01    10:01a    4,094 SPF78F~1.SQL
    09/14/01    10:01a    1,235 SP_ADD~1.SQL
    09/14/01    10:01a    2,035 SP_GET~1.SQL
    09/14/01    10:01a    2,953 SP_TWD~1.SQL
    09/14/01    10:01a    3,196 SP_TWD~2.SQL
15 09/14/01    10:01a    4,430 SP_TWD~3.SQL
    09/14/01    10:01a    6,144 SP_TWD~4.SQL
    09/14/01    10:01a    1,003 TWA2F1~1.SQL
    09/14/01    10:01a    584 TWDB_A~1.SQL
    09/14/01    10:01a    853 TWDB_A~2.SQL
20 09/14/01    10:01a    915 TWDB_A~3.SQL
    09/14/01    10:01a    445 TWDB_G~1.SQL
    09/14/01    10:01a    484 TWDB_G~2.SQL
    09/14/01    10:01a    2,104 TWDB_G~3.SQL
    09/14/01    10:01a    529 TWDB_G~4.SQL
25 09/14/01    10:01a    1,234 TWDB_I~1.SQL
    09/14/01    10:01a    405 TWDB_Q~1.SQL
    09/14/01    10:01a    529 TWDB_R~1.SQL
    09/14/01    10:01a    554 TWDB_R~2.SQL
    09/14/01    10:01a    1,497 TWDB_S~1.SQL
30 09/14/01    10:01a    582 TWDB_U~1.SQL
    09/14/01    10:01a    1,123 TWDB_U~2.SQL
    09/14/01    10:01a    464 VSSVER.SCC
                                30 File(s)
                                54,588 bytes

35 Directory of c:\SPOTNE~1\DATABASE\TWDB\TABLES
    09/18/01    11:25a    <DIR>
    09/18/01    11:25a    <DIR>
    09/14/01    10:01a    479 SPCOBR~1.SQL
    09/14/01    10:01a    793 SPPROM~1.SQL
40 09/14/01    10:01a    562 SPREPO~1.SQL
    09/14/01    10:01a    450 SPUSER~1.SQL
    09/14/01    10:01a    552 TWADMI~1.SQL
    09/14/01    10:01a    389 TWCARR~1.SQL
    09/14/01    10:01a    432 TWCORBR~1.SQL
45 09/14/01    10:01a    548 TWCONT~1.SQL
    09/14/01    10:01a    442 TWEVENT.SQL
    09/14/01    10:01a    434 TWSTATES.SQL
    09/14/01    10:01a    571 TWUSER~1.SQL

```

	09/14/01	10:01a	911 TWUSER~2.SQL
	09/14/01	10:01a	472 TWZIPC~1.SQL
	09/14/01	10:01a	240 VSSVER.SCC
		16 File(s)	7,275 bytes
5	Directory of e:\SPOTNE~1\DATABASE\TWDB\TRIGGERS		
	09/18/01	11:25a	<DIR>
	09/18/01	11:25a	<DIR>
		2 File(s)	0 bytes
10	Directory of e:\SPOTNE~1\DISTRI~1		
	09/18/01	11:25a	<DIR>
	09/18/01	11:25a	<DIR>
	01/25/01	03:07p	6,043 TWDISD~1.JAV
15	01/29/01	04:38p	4,832 TWDISD~2.JAV
	01/25/01	03:06p	2,694 TWDISE~1.JAV
	02/02/01	11:26a	6,215 TWDISF~1.JAV
	02/02/01	11:27a	11,756 TWDISF~2.JAV
	10/02/00	04:55p	5,552 TWDISL~1.JAV
20	09/28/00	09:44a	4,617 TWDISL~2.JAV
	10/05/00	04:49p	6,812 TWDISM~1.JAV
	09/18/00	12:06p	4,826 TWDISP~1.JAV
	09/29/00	10:52a	3,568 TWDISP~2.JAV
	10/03/00	02:14p	6,142 TWDISR~1.JAV
25	08/31/00	09:51a	1,301 TWEVEN~1.JAV
	08/27/00	04:43p	18,238 TWMAIL~1.JAV
	08/28/00	01:13p	3,489 TWTCP~1.JAV
		16 File(s)	86,085 bytes
30	Directory of e:\SPOTNE~1\INTERA~1		
	09/18/01	11:25a	<DIR>
	09/18/01	11:25a	<DIR>
	09/14/01	10:09a	5,062 ALWAYS~1.ASP
	09/14/01	10:09a	2,781 ASSEMB~1.CS
35	09/14/01	10:09a	7,704 CRMREP~1.CSP
	09/14/01	10:09a	936 CRMREP~1.SLN
	09/14/01	10:09a	302 CRMREP~1.VSD
	09/14/01	10:09a	111 CRMREP~1.WEB
	09/14/01	10:09a	7,908 DEFAUL~1.ASP
40	09/14/01	10:09a	4,861 DEFAUL~1.CS
	09/14/01	10:09a	0 DEFAUL~1.RES
	09/18/01	11:25a	<DIR>
	09/14/01	10:09a	76 GLOBAL~1.ASA
	09/14/01	10:09a	760 GLOBAL~1.CS
45	09/14/01	10:09a	0 GLOBAL~1.RES
	09/18/01	11:25a	<DIR>
	09/14/01	10:09a	IMAGES
	09/14/01	10:09a	1,938 LICENS~1.LIC
	09/14/01	10:09a	3,254 LOGINA~1.CS

	09/14/01	10:09a	0 LOGINA~1.RES
	09/14/01	10:09a	3,180 LOGIN~1.ASP
	09/14/01	10:09a	881 MORERE~1.ASP
	09/14/01	10:09a	884 PROCES~1.ASP
5	09/14/01	10:09a	1,189 PROCES~1.CS
	09/14/01	10:09a	0 PROCES~1.RES
	09/14/01	10:09a	8,060 PROCES~2.ASP
	09/14/01	10:09a	29,527 RPTSER~1.ASP
	09/14/01	10:09a	1,520 SMARTV~1.ASP
10	09/14/01	10:09a	249 SMARTV~2.ASP
	09/14/01	10:09a	2,041 SMARTV~3.ASP
	09/14/01	10:09a	2,622 SNCONF~1.CS
	09/14/01	10:09a	76,800 SP51EA~1.RPT
	09/14/01	10:09a	76,288 SP8B69~1.RPT
15	09/14/01	10:09a	70,656 SPFFF7~1.RPT
	09/14/01	10:09a	65,536 SP_TWD~1.RPT
	09/14/01	10:09a	77,824 SP_TWD~2.RPT
	09/14/01	10:09a	77,312 SP_TWD~3.RPT
	09/14/01	10:09a	76,800 SP_TWD~4.RPT
20	09/14/01	10:09a	8,109 TOOLBAR.ASP
	09/14/01	10:09a	1,319 USER.CS
	09/14/01	10:09a	2,485 USERDATA.CS
	09/14/01	10:09a	4,712 WEB~1.CON
		41 File(s)	623,687 bytes
25			
	Directory of e:\SPOTNE~1\INTERA~1\DOCS		
	09/18/01	11:25a	<DIR>
	09/18/01	11:25a	<DIR>
	09/14/01	10:09a	25,600 CRM_AR~1.DOC
30	09/14/01	10:09a	29,696 CRM_DB~1.DOC
	09/14/01	10:09a	34,304 CRM_FU~1.DOC
	09/14/01	10:09a	26,112 CRM_RE~1.DOC
	09/14/01	10:09a	51,200 IR_INS~1.DOC
	09/14/01	10:09a	1,169,408 PRODUC~1.DOC
35	09/14/01	10:09a	21,504 SPOTLI~1.DOC
	09/14/01	10:09a	531 UPDATE~1.TXT
	09/14/01	10:09a	160 VSSVER.SCC
		11 File(s)	1,358,515 bytes
40			
	Directory of e:\SPOTNE~1\INTERA~1\IMAGES		
	09/18/01	11:25a	<DIR>
	09/18/01	11:25a	<DIR>
	09/14/01	10:09a	5,506 100X10~1.JPG
	09/14/01	10:09a	7,164 NESTLE.JPG
45	09/14/01	10:09a	1,767 NESTLE~1.GIF
	09/14/01	10:09a	1,553 NESTLE~2.GIF
	09/14/01	10:09a	6,982 SPOTNET.JPG
	09/14/01	10:09a	5,506 SPOTNE~1.JPG

	09/14/01	10:09a	128 VSSVER.SCC
		9 File(s)	28,606 bytes
	Directory of e:\SPOTNE~1\KAPRON		
5	09/18/01	11:25a	<DIR>
	09/18/01	11:25a	<DIR>
	09/18/01	11:25a	<DIR>
	09/18/01	11:25a	<DIR>
		4 File(s)	0 bytes
10	Directory of e:\SPOTNE~1\KAPRON\ATVEF		
	09/18/01	11:25a	<DIR>
	09/18/01	11:25a	<DIR>
	09/10/00	01:41p	15,681 ATVEF.CPP
15	09/10/00	01:41p	1,848 ATVEF.H
	09/10/00	01:41p	67 CARRIER.SH
	09/10/00	01:41p	587 GTYPE.SH
	09/10/00	01:41p	230 MAKEFILE
	09/10/00	01:41p	1,132 RUN.CPP
20	09/10/00	01:41p	139 RUN.MK
	09/10/00	01:41p	6,864 SENDTR~1.CPP
	09/10/00	01:41p	1,298 TRIGGE~1.SCR
	09/10/00	02:14p	176 VSSVER.SCC
		12 File(s)	28,022 bytes
25	Directory of e:\SPOTNE~1\KAPRON\SERVER		
	09/18/01	11:25a	<DIR>
	09/18/01	11:25a	<DIR>
	09/10/00	01:46p	60 DEMO.SH
30	09/10/00	01:46p	162 MAKEFILE
	09/10/00	01:46p	8,999 SERVER.CPP
	09/10/00	02:14p	80 VSSVER.SCC
		6 File(s)	9,301 bytes
35	Directory of e:\SPOTNE~1\ON-LIN~1		
	09/18/01	11:25a	<DIR>
	09/18/01	11:25a	<DIR>
	11/28/00	09:39p	6,193 TWDATA~1.JAV
	10/04/00	04:41p	1,250 TWDATA~1.TPL
40	11/30/00	05:24p	4,929 TWFILE~1.JAV
	10/04/00	05:07p	919 TWFILE~1.TPL
	09/20/00	05:48p	4,815 TWLOAD~1.JAV
	11/30/00	05:24p	4,335 TWLOAD~2.JAV
	11/28/00	09:06p	6,019 TWLOAD~3.JAV
45	10/05/00	03:35p	1,600 TWLOGIN.TPL
	11/28/00	09:30p	4,842 TWLOGI~1.JAV
	11/30/00	05:23p	24,296 TWMTPT~1.JAV
	11/30/00	05:27p	863 TWNOCA~1.JAV

10/10/00	02:15p	4,745 TWUSER~1.JAV
04/04/01	12:09p	7,534 TWUSER~2.JAV
		72,340 bytes

15 File(s)

5 Directory of e:\SPOTNE~1\REDIRE~1

09/18/01	11:25a	<DIR>	.
09/18/01	11:25a	<DIR>	..
		2 File(s)	0 bytes

10 Directory of e:\SPOTNE~1\TUNER

09/18/01	11:25a	<DIR>	.
09/18/01	11:25a	<DIR>	..
09/18/01	11:25a	<DIR>	BINARY~1
09/18/01	11:25a	<DIR>	JAVAAP~1
09/18/01	11:25a	<DIR>	THINTU~1

5 File(s) 0 bytes

Directory of e:\SPOTNE~1\TUNER\BINARY~1

09/18/01	11:25a	<DIR>	.
20 09/18/01	11:25a	<DIR>	..
04/07/01	06:25p		1,281 COMMAN~1.CPP
04/07/01	06:24p		1,257 COMMAN~1.H
06/05/01	04:56p		12,106 MAINFRM.CPP
05/19/01	04:49p		2,398 MAINFRM.H
25 06/09/01	06:59p		5,706 PREDEF~1.H
04/07/01	07:02p		2,248 README.TXT
09/18/01	11:25a	<DIR>	RES
05/02/01	03:12p		4,596 RESOURCE.H
04/07/01	12:15a		476 STATIC~1.CPP
30 04/07/01	12:14a		56 STATIC~1.H
10/14/00	12:39p		212 STDAFX.CPP
08/26/00	10:36p		1,124 STDAFX.H
05/26/01	03:20p		21,565 TEC22A~1.CPP
05/23/01	10:56p		3,812 TED736~1.H
35 05/26/01	04:40p		10,051 TELEWE~1.RC2
05/23/01	09:21p		2,282 TELEWE~1.H
05/26/01	02:19p		26,565 TELEWE~1.RC
08/20/00	12:09a		407 TELEWE~1.H
05/26/01	03:22p		17,855 TELEWE~2.CPP
40 04/07/01	05:14p		3,236 TELEWE~2.H
05/26/01	03:31p		11,216 TELEWE~3.CPP
04/24/01	03:42p		2,871 TELEWE~3.H
06/09/01	06:52p		20,308 TELEWE~4.CPP
11/04/00	01:10a		3,708 TELEWE~4.H
45 04/07/01	06:09p		1,831 TWPROP~1.CPP
04/07/01	06:04p		1,380 TWPROP~1.H
02/10/01	11:19p		1,179 TWPROP~2.CPP
02/10/01	11:18p		1,498 TWPROP~2.H

09/18/01 11:25a <DIR> TWUGAD~1
09/09/00 09:24p 11,299 WEBBRO~1.CPP
09/09/00 09:24p 3,759 WEBBRO~1.H
33 File(s) 176,282 bytes

5

Directory of e:\SPOTNE~1\TUNER\BINARY~1\RES

	09/18/01	11:25a	<DIR>	
	09/18/01	11:25a	<DIR>	
	04/20/01	02:22p		2,590 ABC_53~1.BMP
10	04/20/01	02:04p		4,878 ABC_73~1.BMP
	04/03/01	04:13p		718 BITMAP12.BMP
	04/03/01	04:13p		718 BITMAP13.BMP
	04/03/01	04:13p		718 BITMAP14.BMP
	04/03/01	04:13p		718 BITMAP15.BMP
15	04/03/01	04:13p		286 BMP00001.BMP
	04/03/01	04:13p		286 BMP00002.BMP
	04/03/01	04:13p		1,334 BMP01.BMP
	04/03/01	04:13p		7,478 BMP02.BMP
	04/03/01	04:13p		7,478 BMP03.BMP
20	04/20/01	02:04p		4,878 BMP04.BMP
	04/20/01	11:35a		4,878 BMP05.BMP
	04/03/01	04:13p		4,878 BMP06.BMP
	04/03/01	04:13p		4,720 BMP07.BMP
	04/03/01	04:13p		4,878 BMP08.BMP
25	04/20/01	11:07a		4,878 BMP09.BMP
	04/03/01	04:13p		4,878 BMP10.BMP
	01/30/01	12:35p		11,054 BMP11.BMP
	01/30/01	12:35p		11,054 BMP11F~1.BMP
	04/03/01	04:13p		11,054 BMP11S~1.BMP
30	04/20/01	11:47a		2,118 BMP12.BMP
	04/20/01	10:50a		4,878 BMP13.BMP
	04/03/01	04:13p		4,878 BMP14.BMP
	04/20/01	02:22p		2,590 BMP15.BMP
	04/20/01	11:46a		874 BMP16.BMP
35	04/20/01	11:09a		2,590 BMP17.BMP
	04/03/01	04:13p		2,590 BMP18.BMP
	04/03/01	04:13p		4,374 BMP19.BMP
	04/20/01	11:44a		2,590 BMP20.BMP
	04/03/01	04:13p		2,590 BMP21.BMP
40	04/03/01	04:13p		2,590 BMP22.BMP
	04/03/01	04:13p		2,432 BMP23.BMP
	04/20/01	11:02a		2,590 BMP24.BMP
	04/03/01	04:13p		2,590 BMP25.BMP
	04/03/01	04:13p		4,878 BMP26.BMP
45	04/03/01	04:13p		2,590 BMP27.BMP
	04/20/01	11:46a		874 E_53X27.BMP
	04/20/01	11:47a		2,118 E_73X50.BMP
	04/20/01	11:02a		2,590 FOODTV~1.BMP

	04/20/01	10:50a	4,878 FOODTV~2.BMP
	04/20/01	11:09a	2,590 FOX_53~1.BMP
	04/20/01	11:07a	4,878 FOX_73~1.BMP
	04/03/01	04:13p	1,334 ICON.ICO
5	04/03/01	04:13p	1,078 IDR_MAIN.ICO
	04/03/01	04:13p	442 LINKSLOG.BMP
	04/20/01	11:44a	2,590 MSNBC_~1.BMP
	04/20/01	11:35a	4,878 MSNBC_~2.BMP
	04/03/01	04:13p	286 NEXTLOGO.BMP
10	04/23/01	04:47p	2,590 NICK_5~1.BMP
	04/23/01	04:18p	4,878 NICK_7~1.BMP
	04/03/01	04:13p	286 PREVIOUS.BMP
	04/03/01	04:13p	442 SETUPLOG.BMP
	04/03/01	04:13p	1,726 SMALLLOG.BMP
15	04/23/01	04:47p	2,590 SNGAME~1.BMP
	04/20/01	11:29a	4,878 SNGAME~2.BMP
	04/23/01	04:47p	2,590 SNKIDS~1.BMP
	04/23/01	04:18p	4,878 SNKIDS~2.BMP
	04/23/01	04:47p	2,590 SNSPOR~1.BMP
20	04/20/01	11:26a	4,878 SNSPOR~2.BMP
	04/23/01	04:47p	2,590 SNWOME~1.BMP
	04/23/01	04:18p	4,878 SNWOME~2.BMP
	04/23/01	04:47p	2,590 SONY_5~1.BMP
	04/23/01	04:47p	4,878 SONY_7~1.BMP
25	04/03/01	04:13p	2,574 SPOTNETB.BMP
	04/03/01	04:13p	2,574 SPOTNE~1.BMP
	04/03/01	04:13p	1,834 SPOTNE~2.BMP
	04/03/01	04:13p	20,594 SPOTNE~3.BMP
	04/03/01	04:13p	20,594 SPOTNE~4.BMP
30	04/03/01	04:13p	1,078 TELEWE~1.ICO
	04/03/01	04:13p	407 TELEWE~1.RC2
	04/03/01	04:13p	1,078 TELEWE~2.ICO
	04/24/01	04:10p	1,088 VSSVER.SCC
	04/23/01	04:47p	2,590 WB_53X27.BMP
35	04/23/01	04:47p	4,878 WB_73X50.BMP

77 File(s)
277,181 bytes

Directory of e:\SPOTNE~1\TUNER\BINARY~1\TWUGAD~1

40	09/18/01	11:25a	<DIR>	.
	09/18/01	11:25a	<DIR>	..
	09/18/01	11:25a	<DIR>	DEBUG
	04/03/01	04:13p		167 MSSCCPRJ.SCC
	03/03/01	03:40p		1,227 README.TXT
	09/18/01	11:25a	<DIR>	RELEASE
45	09/18/01	11:25a	<DIR>	STAGIN~1
	09/18/01	11:25a	<DIR>	STAGIN~2
	03/03/01	03:40p		296 STDAFX.CPP
	03/03/01	03:40p		773 STDAFX.H

	06/09/01	06:54p	1,256 TWUGAD~1.CPP
	04/03/01	04:42p	7,404 TWUGAD~1.DSP
	04/29/01	01:59p	1,658 TWUGAD~1.PLG
	06/09/01	06:54p	112 VSSVER.SCC
5		14 File(s)	12,893 bytes
Directory of e:\SPOTNE~1\TUNER\BINARY~1\TWUGAD~1\DEBUG			
	09/18/01	11:25a	<DIR>
	09/18/01	11:25a	<DIR>
10	06/09/01	06:55p	55,057 STDAFX.OBJ
	06/09/01	06:55p	3,706 TWUGAD~1.OBJ
	06/09/01	06:55p	1,960,604 TWUGAD~1.PCH
	06/09/01	06:55p	377,856 TWUGAD~1.PDB
	06/09/01	06:55p	82,944 VC60.IDB
15	06/09/01	06:55p	176,128 VC60.PDB
		8 File(s)	2,656,295 bytes
Directory of e:\SPOTNE~1\TUNER\BINARY~1\TWUGAD~1\RELEASE			
	09/18/01	11:25a	<DIR>
20	09/18/01	11:25a	<DIR>
	06/09/01	06:55	228 STDAFX.OBJ
	06/09/01	06:55p	1,388 TWUGAD~1.OBJ
	06/09/01	06:55p	1,830,052 TWUGAD~1.PCH
	06/09/01	06:55p	41,984 VC60.IDB
25		6 File(s)	1,873,652 bytes
Directory of e:\SPOTNE~1\TUNER\BINARY~1\TWUGAD~1\STAGIN~1			
	09/18/01	11:25a	<DIR>
	09/18/01	11:25a	<DIR>
30	06/09/01	07:01p	228 STDAFX.OBJ
	06/09/01	07:01p	1,388 TWUGAD~1.OBJ
	06/09/01	07:01p	1,830,052 TWUGAD~1.PCH
	06/09/01	07:01p	41,984 VC60.IDB
35		6 File(s)	1,873,652 bytes
Directory of e:\SPOTNE~1\TUNER\BINARY~1\TWUGAD~1\STAGIN~2			
	09/18/01	11:25a	<DIR>
	09/18/01	11:25a	<DIR>
	06/09/01	06:56p	55,105 STDAFX.OBJ
40	06/09/01	06:56p	3,754 TWUGAD~1.OBJ
	06/09/01	06:55p	1,960,604 TWUGAD~1.PCH
	06/09/01	06:55p	377,856 TWUGAD~1.PDB
	06/09/01	06:56p	82,944 VC60.IDB
	06/09/01	06:55p	176,128 VC60.PDB
45		8 File(s)	2,656,391 bytes
Directory of e:\SPOTNE~1\TUNER\JAVAAP~1			
	09/18/01	11:25a	<DIR>

	09/18/01	11:25a	<DIR>	
	07/03/01	10:04a		3,132 APPLT~1.JAV
	07/03/01	10:42a		1,925 AUTO.AU
	08/09/01	11:06a		157 AUTOGE~1.HTM
5	07/03/01	10:04a		3,483 CARRIE~1.JAV
	07/03/01	10:42a		2,417 CHANNEL.AU
	08/21/01	04:31p		6,522 CLIENT~1.JAV
	07/03/01	10:04a		5,310 IMAGEB~1.JAV
	07/03/01	10:42a		756 LINK.AU
10	08/22/01	01:42p		11,539 LINKPA~1.JAV
	08/15/01	04:06p		3,356 LOGOPA~1.JAV
	07/03/01	10:42a		1,222 NICK
	08/21/01	04:17p		1,346 TRIGGE~1.JAV
	08/22/01	01:42p		1,116 TUNERA~1.CDB
15	08/21/01	01:49p		12,833 TUNERA~1.JAV
	09/10/01	04:43p		118,976 TUNERA~1.VE2
	08/22/01	02:21p		129,113 TUNERA~1.VEP
	07/03/01	01:43p		3,632 TUNERA~1.VPJ
	08/09/01	12:04p		4,115 TW-APP~1.HTM
20	08/09/01	12:07p		539 TWOD13~1.HTM
			21 File(s)	311,489 bytes

Directory of e:\SPOTNE~1\TUNER\THINTU~1

	09/18/01	11:25a	<DIR>	
25	09/18/01	11:25a	<DIR>	
	05/14/01	10:28a		395 CARRIE~1.JAV
	08/13/01	03:10p		1,233 INDEX~1.HTM
	08/16/01	02:55p		3,261 MID_LE~1.HTM
	08/13/01	12:16p		1,175 MID_RI~1.HTM
30	08/22/01	01:07p		1,787 TRIGGE~1.JAV
	08/13/01	01:41p		369 TUNER_~1.HTM
	08/10/01	01:35p		356 TUNER_~2.HTM
	08/15/01	12:05p		652 TUNER_~3.HTM
	08/13/01	01:41p		336 TUNER_~4.HTM
35	09/06/01	02:09p		762 TWPAGE~1.CDB
	06/01/01	09:23a		4,635 TWPAGE~1.JAV
	08/22/01	01:03p		11,846 TWPAGE~2.JAV
	08/16/01	03:02p		4,807 TWPAGE~3.JAV
			File(s)	31,614 bytes

Directory of e:\SPOTNE~1\TWWEBS~1

	09/18/01	11:25a	<DIR>	
	09/18/01	11:25a	<DIR>	
	09/18/01	11:25a	<DIR>	SRC
45	03/20/01	04:32p		13,978 TWSERV~1.JAV
	03/20/01	04:32p		35 TWSERV~1.PRO
	03/20/01	04:32p		4,431 TWWEBR~1.JAV
			6 File(s)	18,444 bytes

Directory of e:\SPOTNE~1\TWWEBS~1\SRC

	09/18/01	11:25a	<DIR>	.
	09/18/01	11:25a	<DIR>	..
5	09/18/01	11:25a	<DIR>	COM
			33 File(s)	0 bytes

Directory of e:\SPOTNE~1\TWWEBS~1\SRC\COM

	09/18/01	11:25a	<DIR>	.
10	09/18/01	11:25a	<DIR>	..
	09/18/01	11:25a	<DIR>	SPOTNET
			3 File(s)	0 bytes

Directory of e:\SPOTNE~1\TWWEBS~1\SRC\COM\SPOTNET

15	09/18/01	11:25a	<DIR>	.
	09/18/01	11:25a	<DIR>	..
	09/18/01	11:25a	<DIR>	SHARED
	09/18/01	11:25a	<DIR>	TW
			4 File(s)	0 bytes

Directory of e:\SPOTNE~1\TWWEBS~1\SRC\COM\SPOTNET\SHARED

	09/18/01	11:25a	<DIR>	.
	09/18/01	11:25a	<DIR>	..
	09/12/00	12:25p		1,872 TWBIGS~1.JAV
25	09/01/00	12:36p		4,258 TWDISL~1.JAV
	03/06/01	07:04p		5,111 TWEVEN~1.JAV
	03/01/01	04:02p		2,270 TWSEVE~1.JAV
	09/11/00	03:07p		4,735 TWWEBL~1.JAV
	03/01/01	04:03p		3,230 TWWEBP~1.JAV
30	05/04/01	11:11a		128 VSSVER.SCC
			9 File(s)	21,604 bytes

Directory of e:\SPOTNE~1\TWWEBS~1\SRC\COM\SPOTNET\TW

	09/18/01	11:25a	<DIR>	.
35	09/18/01	11:25a	<DIR>	..
	08/17/00	07:06p		383 DISTRI~1.PRO
	08/04/00	03:58p		3,223 SURVIVOR.TXT
	08/17/00	07:31p		1,411 TWDISD~1.JAV
	08/17/00	07:27p		1,804 TWDISD~2.JAV
40	08/17/00	07:21p		1,893 TWDISL~1.JAV
	08/17/00	07:20p		4,261 TWDISL~2.JAV
	08/17/00	07:29p		4,265 TWDISM~1.JAV
	08/17/00	07:29p		4,843 TWDISP~1.JAV
	08/17/00	07:19p		1,753 TWDISP~2.JAV
45	08/17/00	02:59p		1,479 TWDISR~1.JAV
	08/17/00	08:22p		1,193 TWEVEN~1.JAV
	08/17/00	07:39p		3,316 TWTCPR~1.JAV
	08/25/00	02:57p		8,593 TY.JAR

M-11052 US
787733 v1

15 File(s)	38,417 bytes
Total Files Listed:	
430 File(s)	12,220,410 bytes
	0 bytes free